



Jet Propulsion Laboratory
California Institute of Technology

THE EARTH SYSTEM GRID FEDERATION AS A TESTBED FOR GLOBAL DISTRIBUTED DATA ANALYTICS

**WORKSHOP ON
REMOTE SENSING, UNCERTAINTY QUANTIFICATION, AND A THEORY OF DATA SYSTEMS
CALTECH, PASADENA (CA)
FEBRUARY 2018**

**LUCA CINQUINI
JET PROPULSION LABORATORY, CALIFORNIA INSTITUTE OF TECHNOLOGY**

**© 2018. ALL RIGHTS RESERVED.
JPL UNLIMITED RELEASE CLEARANCE NUMBER: #17-5659**

PART 1

INTRODUCTION TO THE EARTH SYSTEM GRID FEDERATION

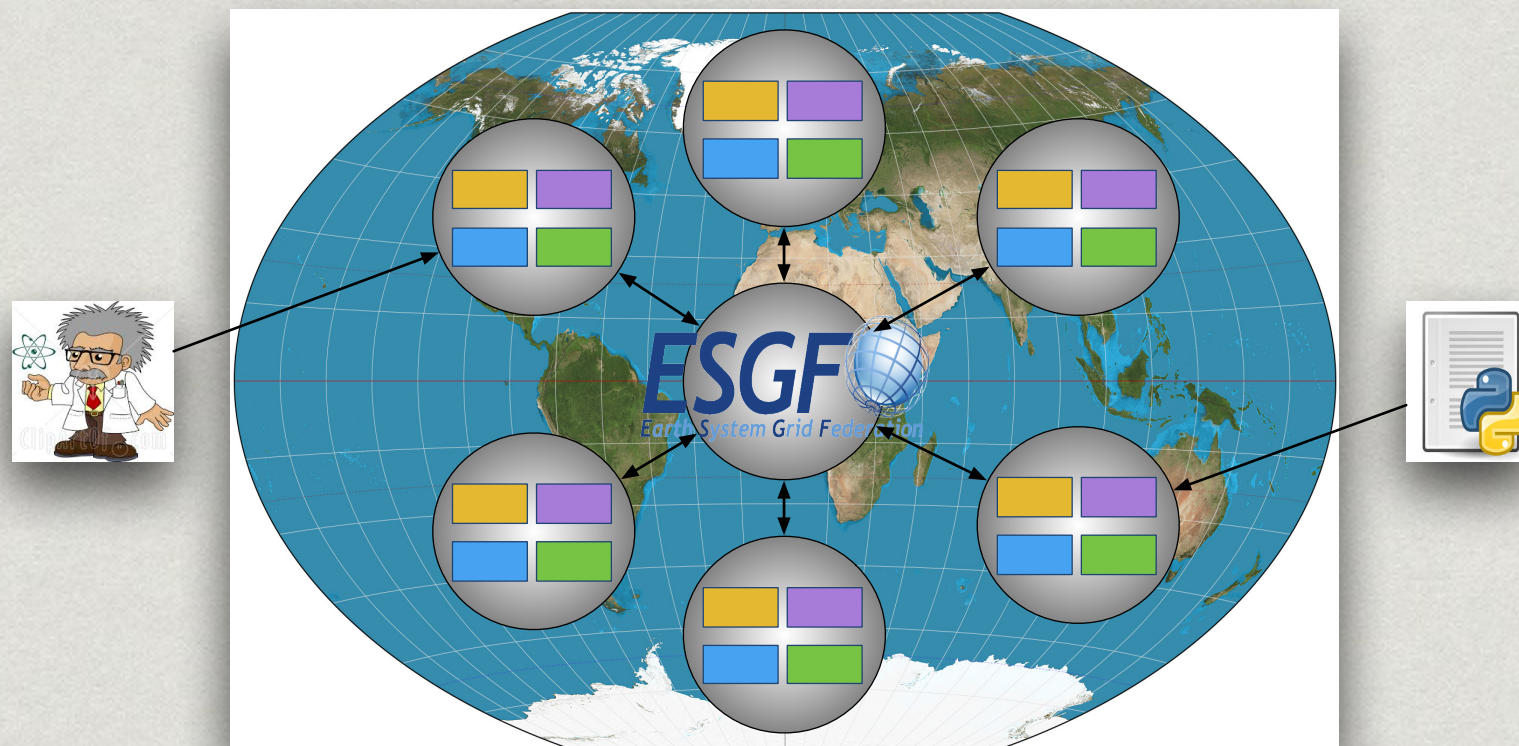
ESGF Overview

- * ESGF is an international collaboration of climate centers working together to manage and provide access to climate data - models and observations
- * Started more than a decade ago, now the world premier technology infrastructure in support of climate science
- * Spanning several tens of institutions in Europe, North America, Australia and Asia
- * Funding from DOE, NASA (U.S.), Copernicus (EU), NCI (Australia), CRIM (Canada)
- * Winner of the 2017 “R&D 100 Award” - prestigious conference that every year recognizes the top 100 most innovative products in software, science and technology



System Architecture

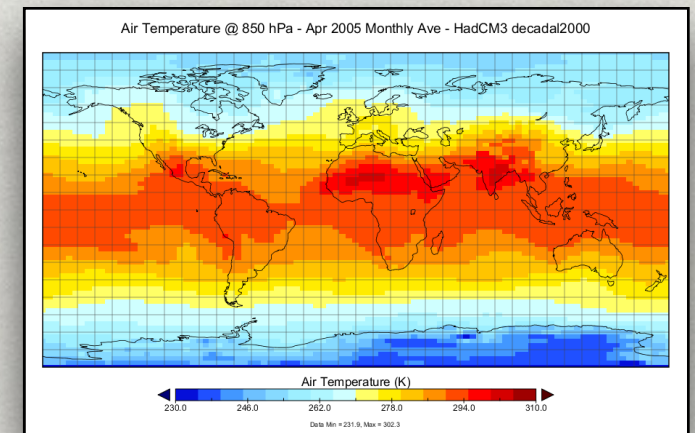
- * ESGF is a system of distributed and federated Nodes that host data and services
- * Distributed: data and metadata are published, stored and served from multiple Nodes across the world
- * Federated: Nodes interoperate because of the adoption of common services, protocols and APIs, and the establishment of mutual trust relationships
- * A client (browser or program) can start from any Node in the federation and discover, download and analyze data from multiple locations as if they were stored in a single central archive



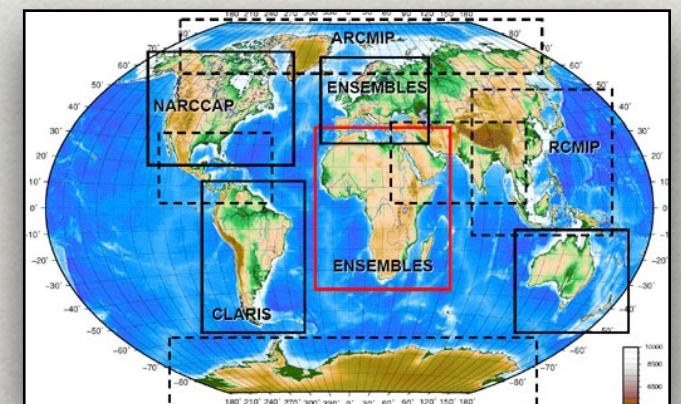
Current Data Holdings

- * ESGF hosts some of the most prominent data collections for climate change research:
 - * CMIP3, CMIP5 (“Coupled Model Inter-Comparison Project”): output of global climate model used for periodic IPCC assessment reports on climate change
 - * CORDEX (“COordinated Regional climate Downscaling EXperiment”): output of regional climate models, grouped by domain (N. America, Europe, Antarctica, etc.)
 - * Obs4MIPs (“Observations for Model Inter-Comparison”): observational data from NASA, ESA, etc. formatted to look like climate model output
 - * Ana4MIPS (“ReAnalysis for Model Inter-Comparison”): re-analysis data formatted like model output
 - * Many other MIPs: TAMIP, GeoMIP, DCMIP, ...
- * The World Climate Research Program (WCRP) has recommended that ESGF infrastructure be supported operationally, and that all future MIPs follow the CMIP5 process and standards (October '12)

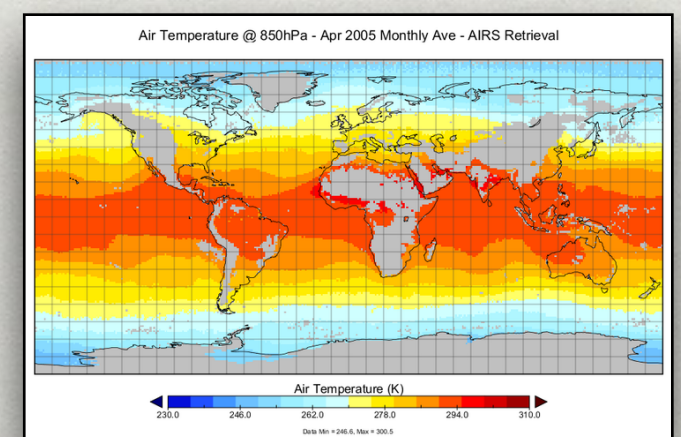
HadCM3



CORDEX



Obs4MIPs



Future Data Holdings

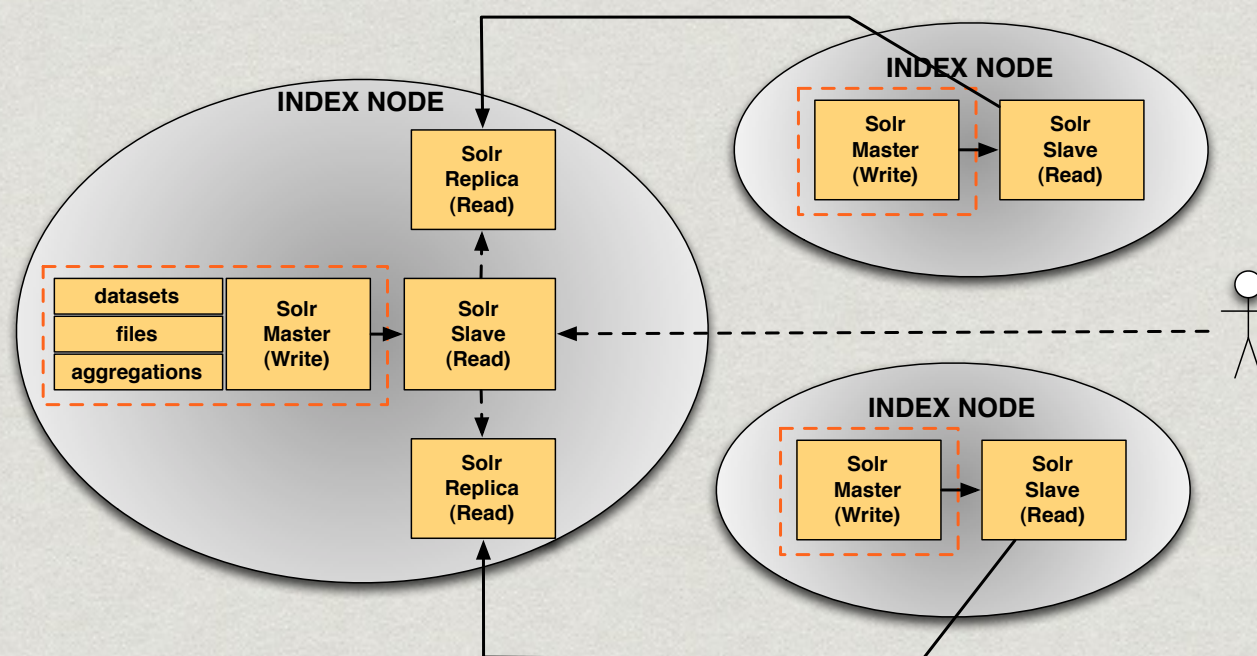
- * ESGF is preparing for a massive increase in its data holdings (10x in the next 3 years):
 - * CMIP6 - starting in early 2018 and terminating in mid 2019
 - * Models runs by ~30 modeling centers around the world
 - * Approximately 25-40 PB of uncompressed primary output, replicated at 4 sites
 - * Supporting CMIP6 will require enhanced scalability and new services for metadata, errata, persistent identifiers
 - * ESGF is holding a series of “Data and Services” challenges leading to a formal release of CMIP6 data on June 1st, 2018
 - * Obs4MIPs - expected ~200 more data collections from NASA, NOAA and European agencies
 - * Dozens of additional MIPs expected to leverage ESGF infrastructure

PART 2

ESGF DATA ACCESS AND ANALYTICAL CAPABILITIES

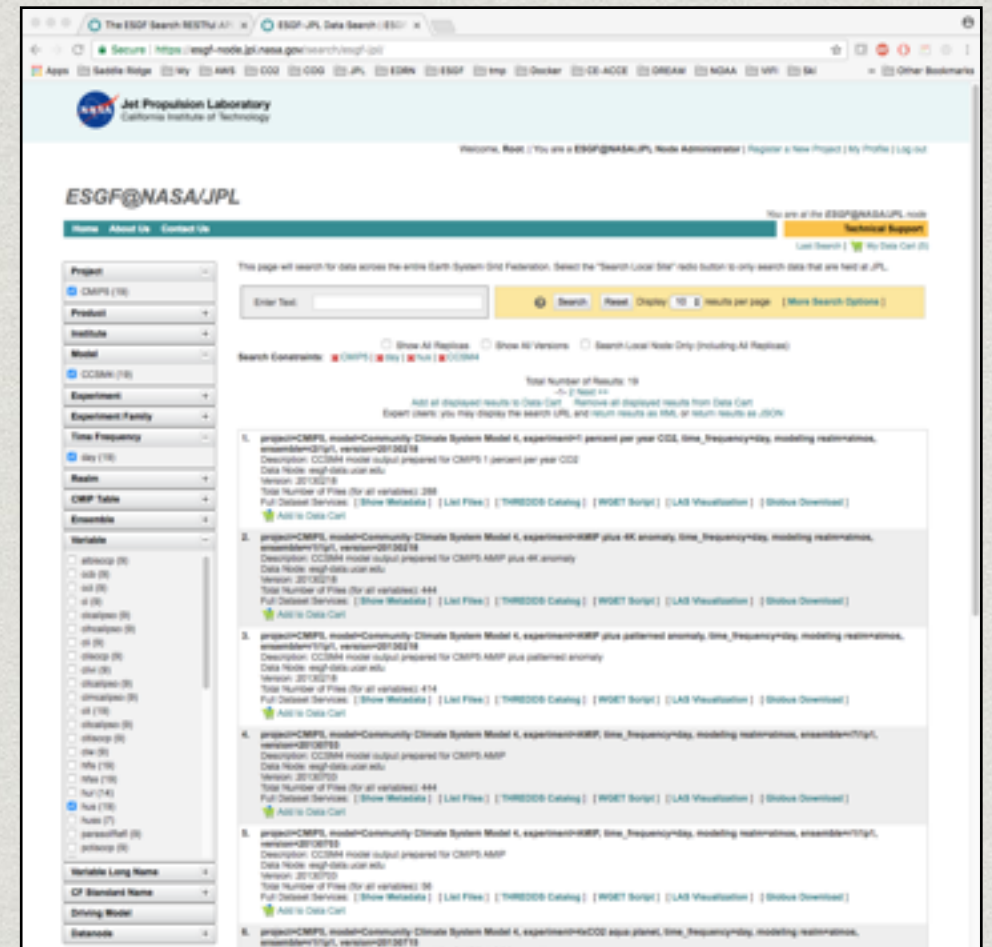
Federated Search

- * ESGF features a state-of-the-art federated search based on Apache Solr, including advanced features as distributed searches and replication
- * Metadata are stored on separate catalogs at multiple sites, yet a search initiated at any site is able to find results throughout the federation
- * Each site runs at least 2 Solr instances: one master Solr to publish metadata (“write”) and one slave Solr to search (“read”)
- * Optionally, a site can choose to replicate some or all of the remote catalogs to improve search performance



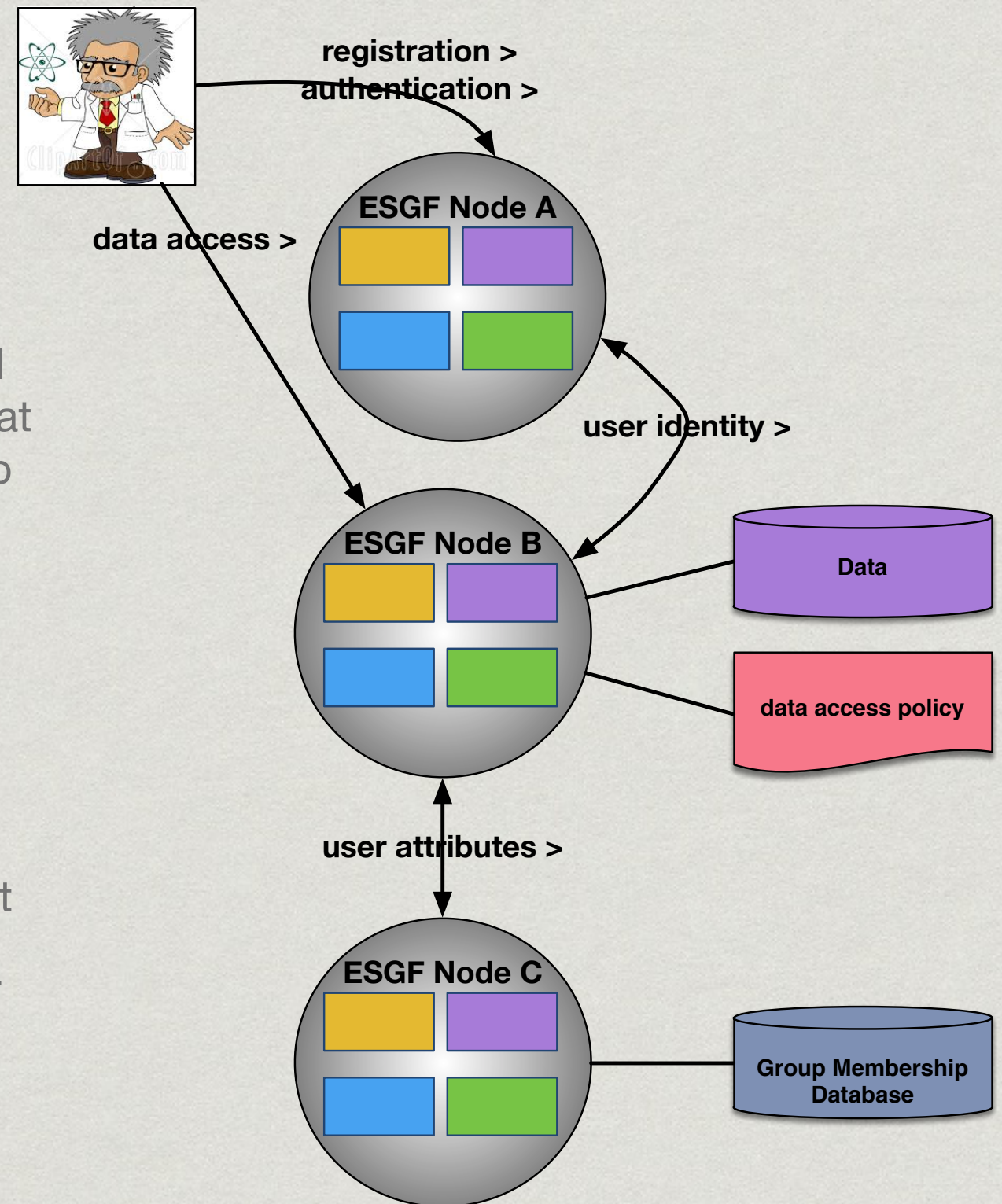
Search API

- ✳ ESGF exposes a RESTful API to query its distributed metadata catalogs:
 - ✳ Available to any HTTP client, including the front-end web portals
 - ✳ The HTTP client can start the query from any index node, results will span the whole federation
 - ✳ Query syntax supports both free text and climate-specific keywords (aka facets)
 - ✳ Results returned as XML or JSON, possibly paginated
- ✳ Query for CMIP5 model output from a specific model that contains daily humidity:
 - ✳ https://esgf-node.jpl.nasa.gov/esg-search/search/?project=CMIP5&variable=hus&model=CCSM4&time_frequency=day
- ✳ See: https://www.earthsystemcog.org/projects/cog/esgf_search_restful_api



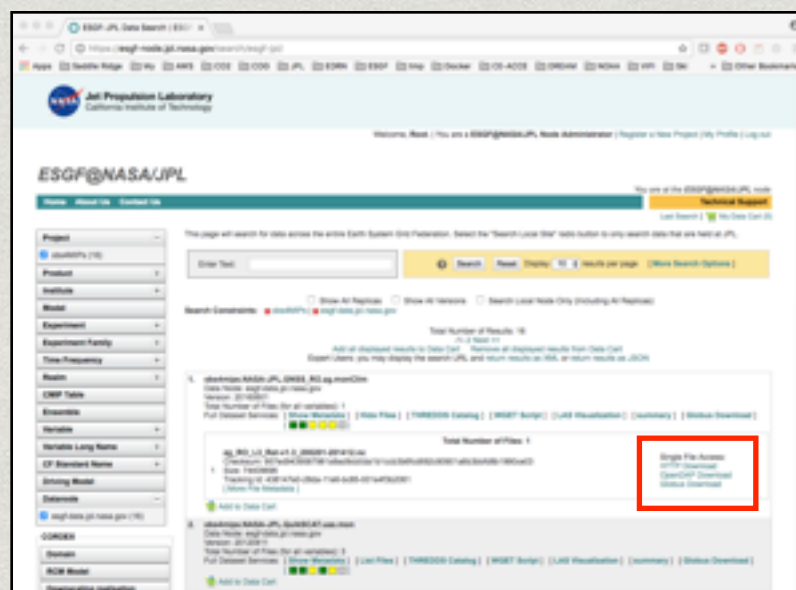
Authentication & Authorization

- * ESGF features a federated authentication and authorization model:
 - * Authentication: users can register at any Node, they are assigned an OpenID which they can use to authenticate anywhere in the federation
 - * Authorization: each Node has complete control over local resources by establishing policies that match group of resources to the required group membership for specific operations - tuple: (resource, group, policy)
- * Despite our best efforts, security remains an obstacle when users want to access data
- * Upcoming security improvements:
 - * OpenID 2.0 —> OAuth2.0 and OpenID Connect
 - * Group membership —> free data download for major data collections
 - * MyProxy —> SLCS server to request an X.509 certificate via HTTP for some operations (data publishing and data processing)

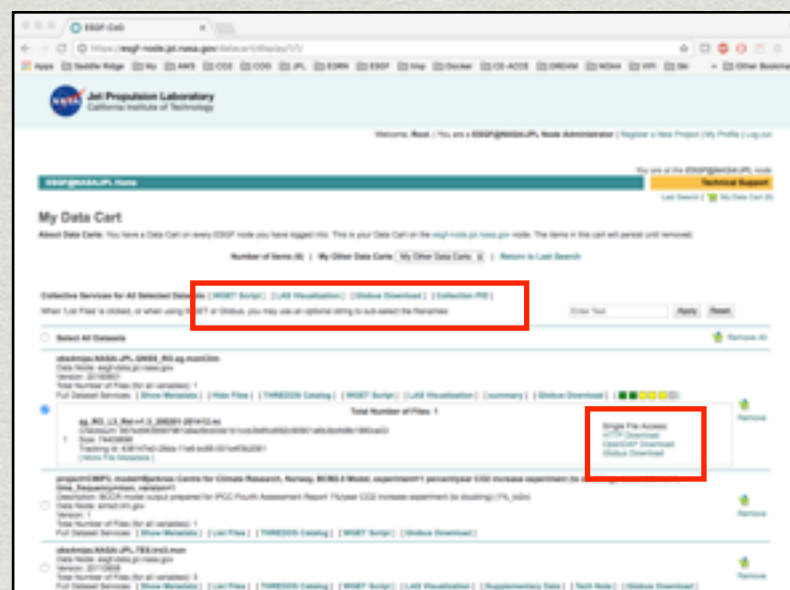


Data Download

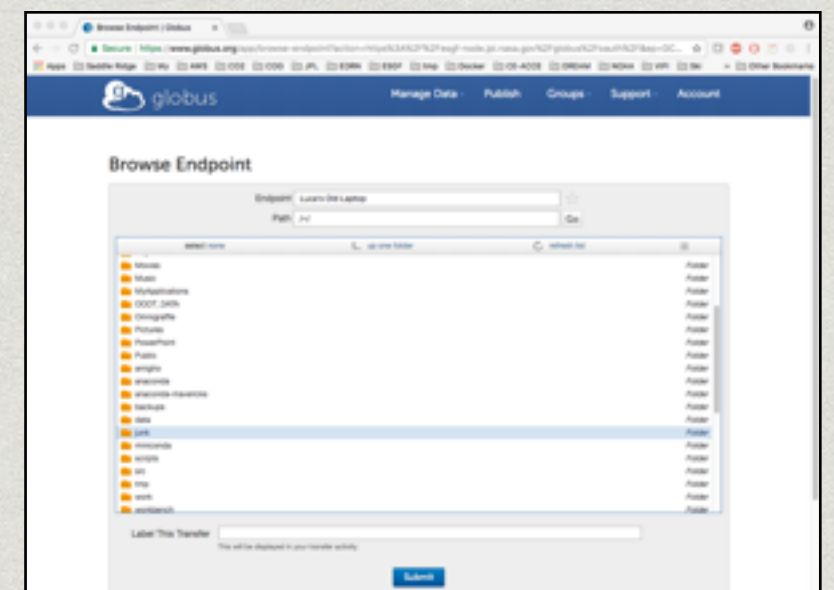
- ✳ ESGF supports several methods to download data from the system:
 - ✳ Using a browser to initiate a single file download via HTTP
 - ✳ Using a browser to generate a wget script to download a large number of files via HTTP
 - ✳ Using a browser to initiate a Globus download through GridFTP
 - ✳ Using OpenDAP API to select specific variables, sub-set by space and time
 - ✳ Using esgfpv client to search and download full files
- ✳ Currently all methods require authentication via OpenID or X.509 cert + group membership



Search Results



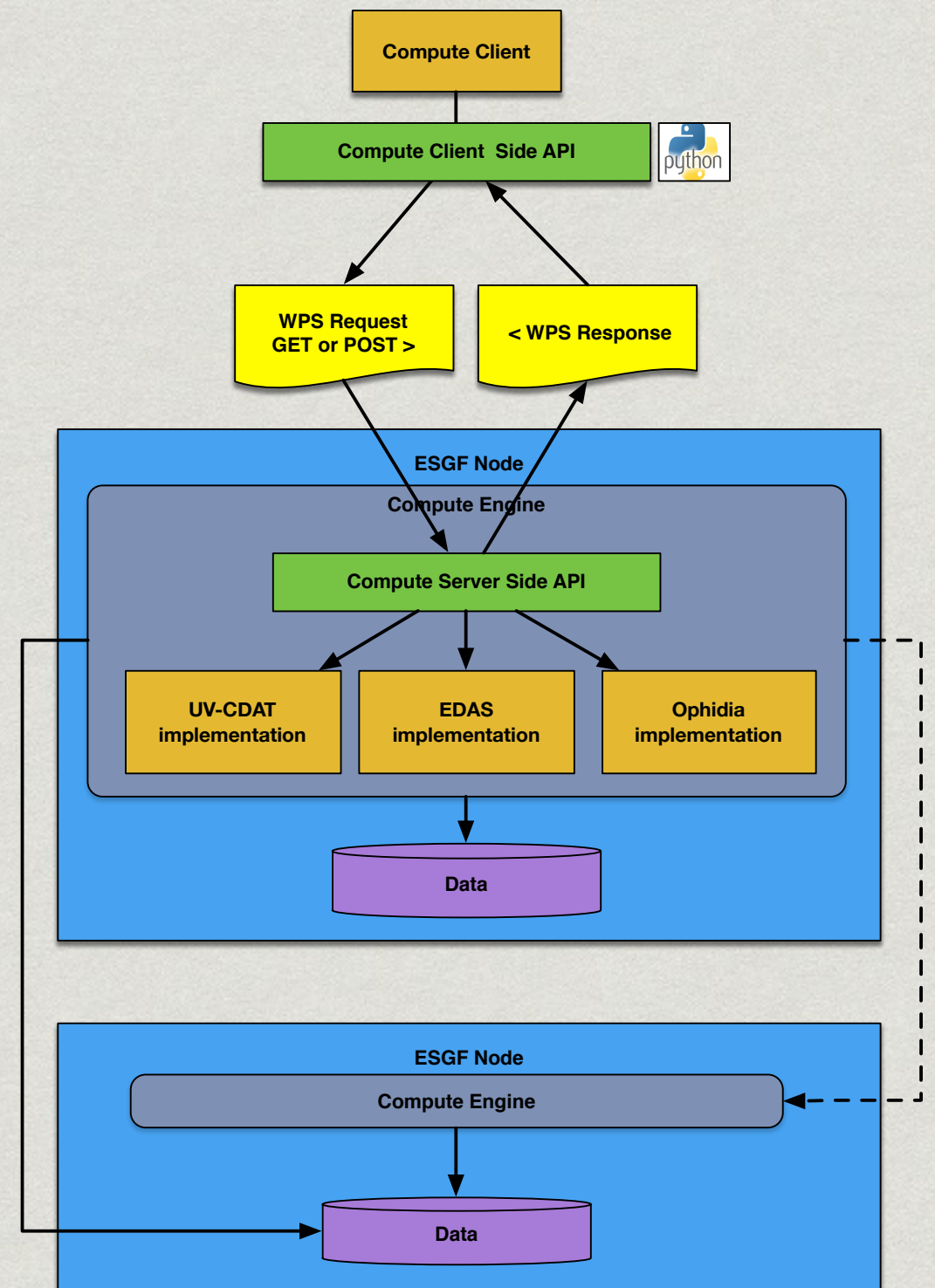
Data Cart



Globus Download

Remote Computing

- * ESGF is working at enabling server-side computing i.e. moving the computation to the data
 - * Motivated by ever large size of data archives - impossible to download to a central location
 - * Problem is made even more complex by the distributed nature of the archives
- * Compute architecture under development is based on the client-server paradigm:
 - * Each ESGF Node deploys a Compute processing server
 - * A client (program or web portal) makes HTTP request to one server which optionally retrieves data from other servers through OpenDAP
 - * Currently operations can be performed at one Node only

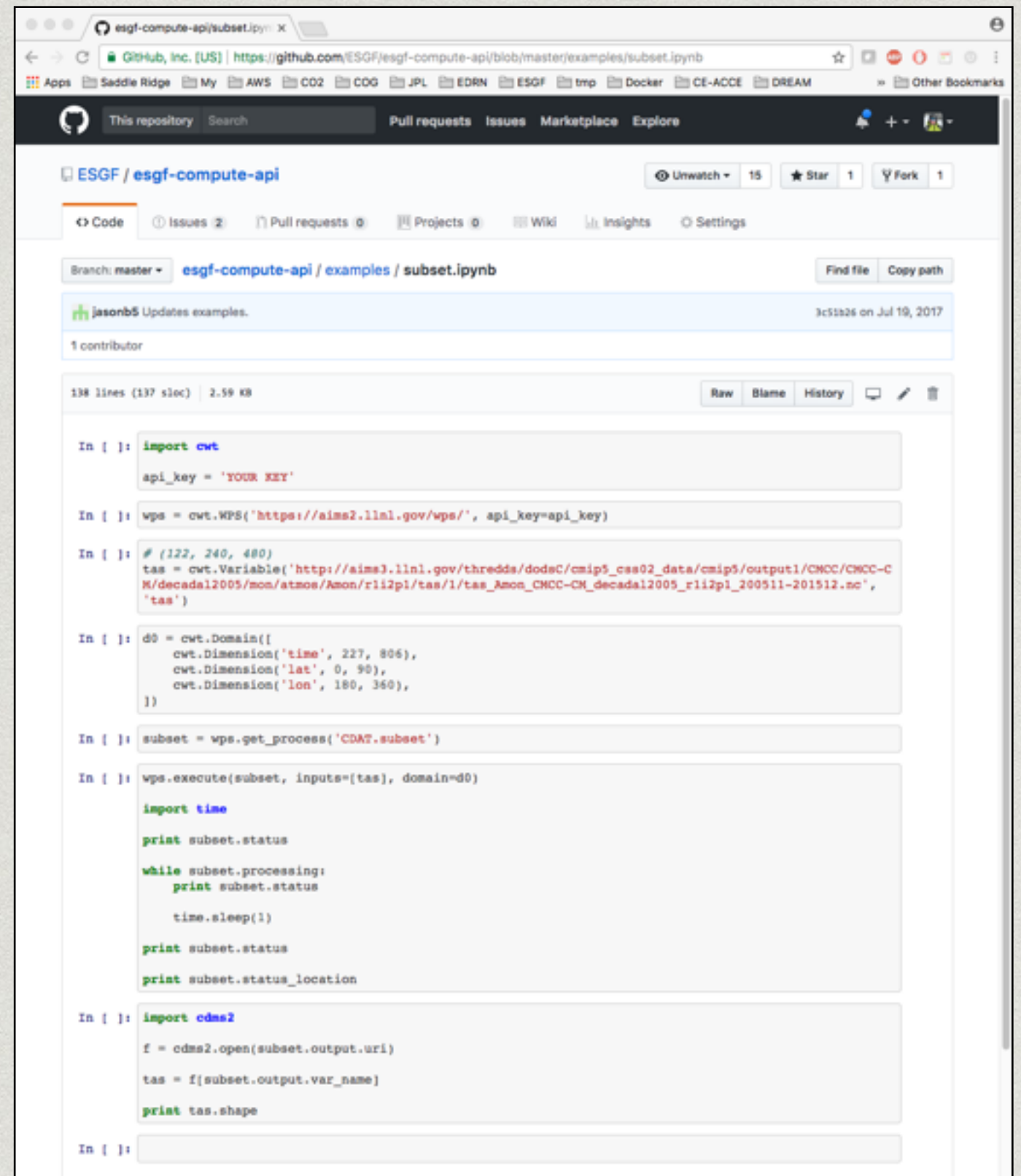


Remote Computing: Server Side

- * ESGF defined a server-side computing API that conforms to the OGC/WPS standard
 - * GET: `http://hostname/wps?service=wps&request=getcapabilities`
 - * GET: `http://hostname/wps?service=wps&request=describeprocess&identifier=grid`
 - * GET: `http://hostname/wps?service=wps&request=execute&identifier=grid&datainputs=dataset=OCO2;algorithm=simple_averaging`
 - * POST: supports more complex requests with XML payload
- * 3 different back-end implementations of server-side API:
 - * UV-CDAT (LLNL): subset, aggregate, regrid, min, max, supports curvilinear grids
 - * EDAS (NCCS/GSFC): parallelized (Spark) subset, aggregate, regrid, ensemble (mul, diff, min, max, ave, sum), and reduction (mul, diff, min, max, ave, sum, rms), supports composition of canonical operations into workflows
 - * Ophidia (CMCC): subset and reduction (max,min)
- * Django application packaged as Docker container brokers request to available(s) back-end implementation
- * Status: test servers deployed at LLNL, NCCS, currently being integrated with ESGF software stack
- * Currently no provision for running custom analytics
- * Documentation: <https://github.com/ESGF/esgf-compute-wps>

Remote Computing: Client Side

- ✳ ESGF developed a client-side Python library to facilitate the process of creating, sending and monitoring compute requests
- ✳ Several examples available as Jupyter notebooks
- ✳ Requires authentication:
 - ✳ User first logs onto web portal with OpenID, password to obtain an authorization token
 - ✳ Authorization token is used as part of client toolkit
- ✳ Documentation: <https://github.com/ESGF/esgf-compute-api>



The screenshot shows a web browser displaying the GitHub repository for `ESGF/esgf-compute-api`. The page is for a Jupyter notebook file named `subset.ipynb` located in the `examples` directory. The notebook content is visible, showing Python code for interacting with the ESGF API. The code includes imports for `cwt` and `cdms2`, setting an API key, creating a WPS object, defining a domain, executing a subset request, and finally opening the output file with `cdms2` to print its shape.

```
In [ ]: import cwt
        api_key = 'YOUR KEY'

In [ ]: wps = cwt.WPS('https://aims2.llnl.gov/wps/', api_key=api_key)

In [ ]: # (122, 240, 480)
        tas = cwt.Variable('http://aims2.llnl.gov/thredds/dodsC/cmip5_data/cmip5/output1/CMCC/CMCC-CM/decadal2005/mom/atmos/Amon/r1i2p1/tas/1/tas_Amon_CMCC-CM_decadal2005_r1i2p1_200511-201512.nc',
                           'tas')

In [ ]: d0 = cwt.Domain([
        cwt.Dimension('time', 227, 806),
        cwt.Dimension('lat', 0, 90),
        cwt.Dimension('lon', 180, 360),
        ])

In [ ]: subset = wps.get_process('CDAT.subset')

In [ ]: wps.execute(subset, inputs=[tas], domain=d0)

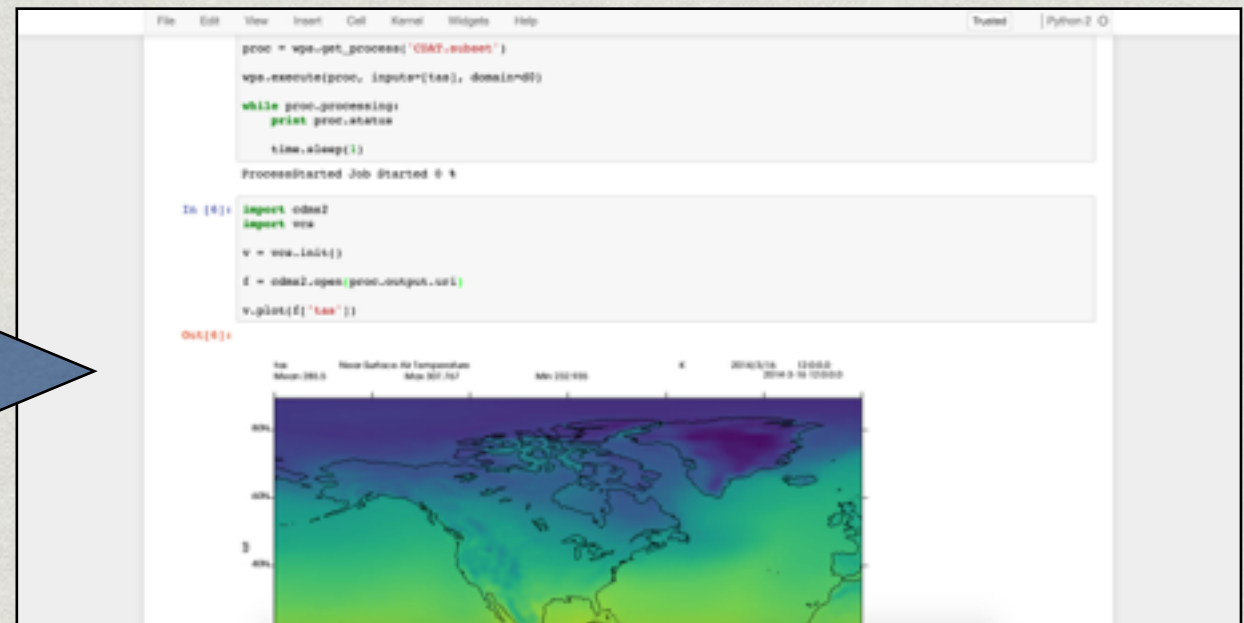
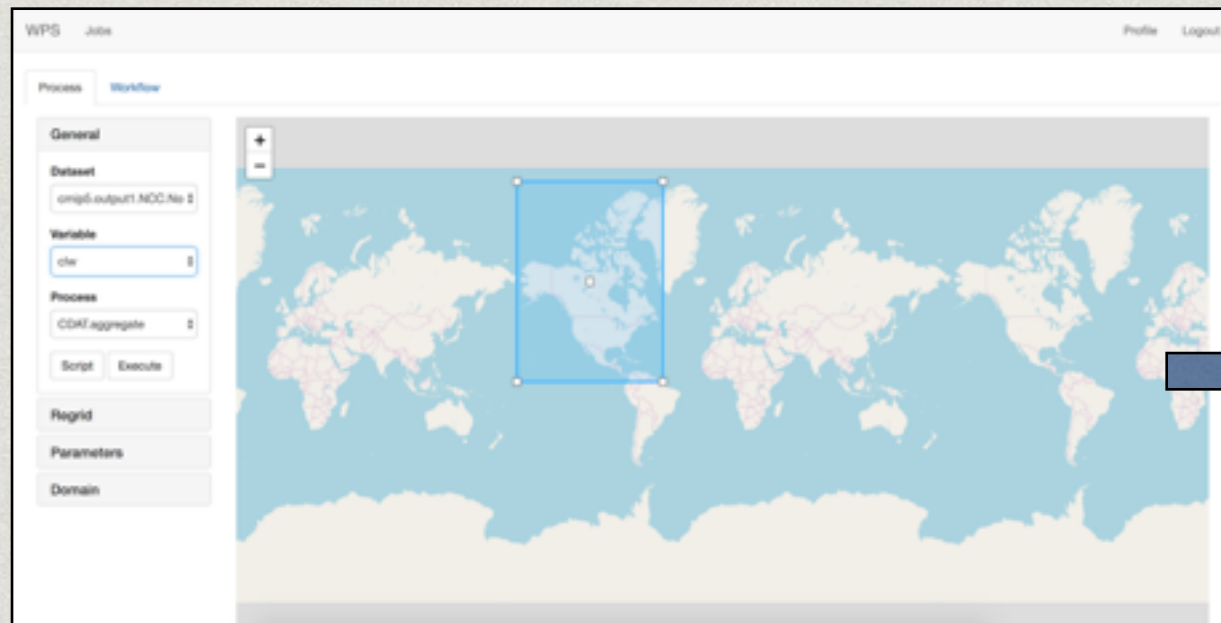
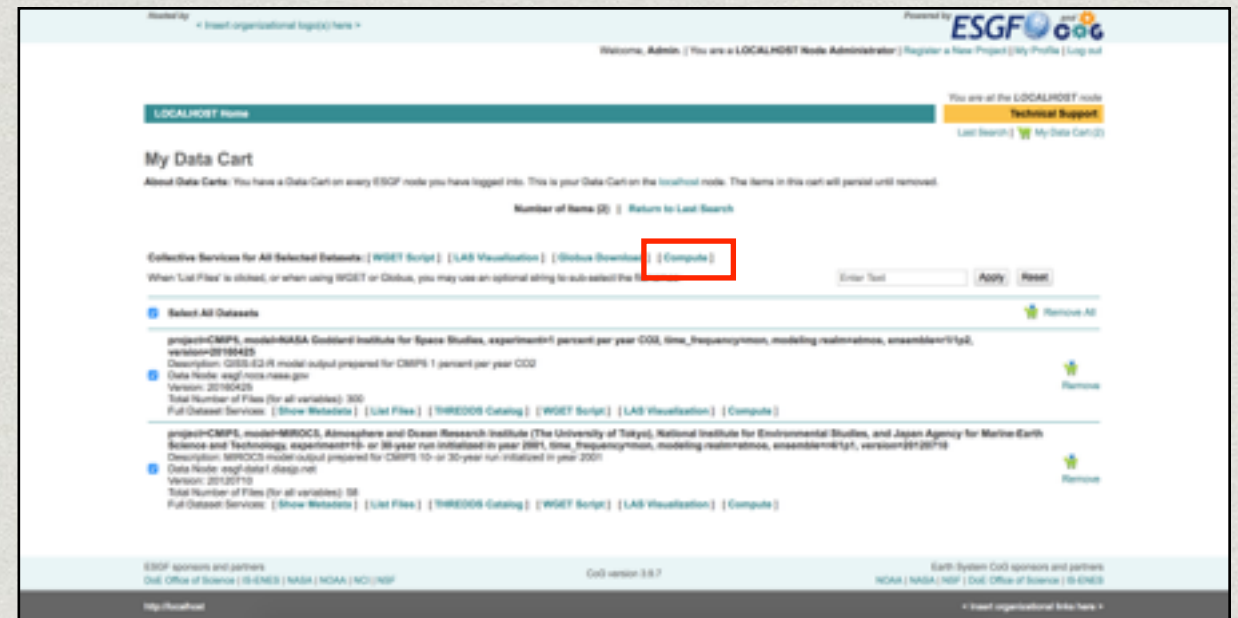
        import time
        print subset.status
        while subset.processing:
            print subset.status
            time.sleep(1)
        print subset.status
        print subset.status_location

In [ ]: import cdms2
        f = cdms2.open(subset.output.uri)
        tas = f[subset.output.var_name]
        print tas.shape

In [ ]:
```


Remote Computing: Web Client

- ✳ Also available: web user interface to formulate and send requests to the remote Compute engine
- ✳ Integrated with standard ESGF UI

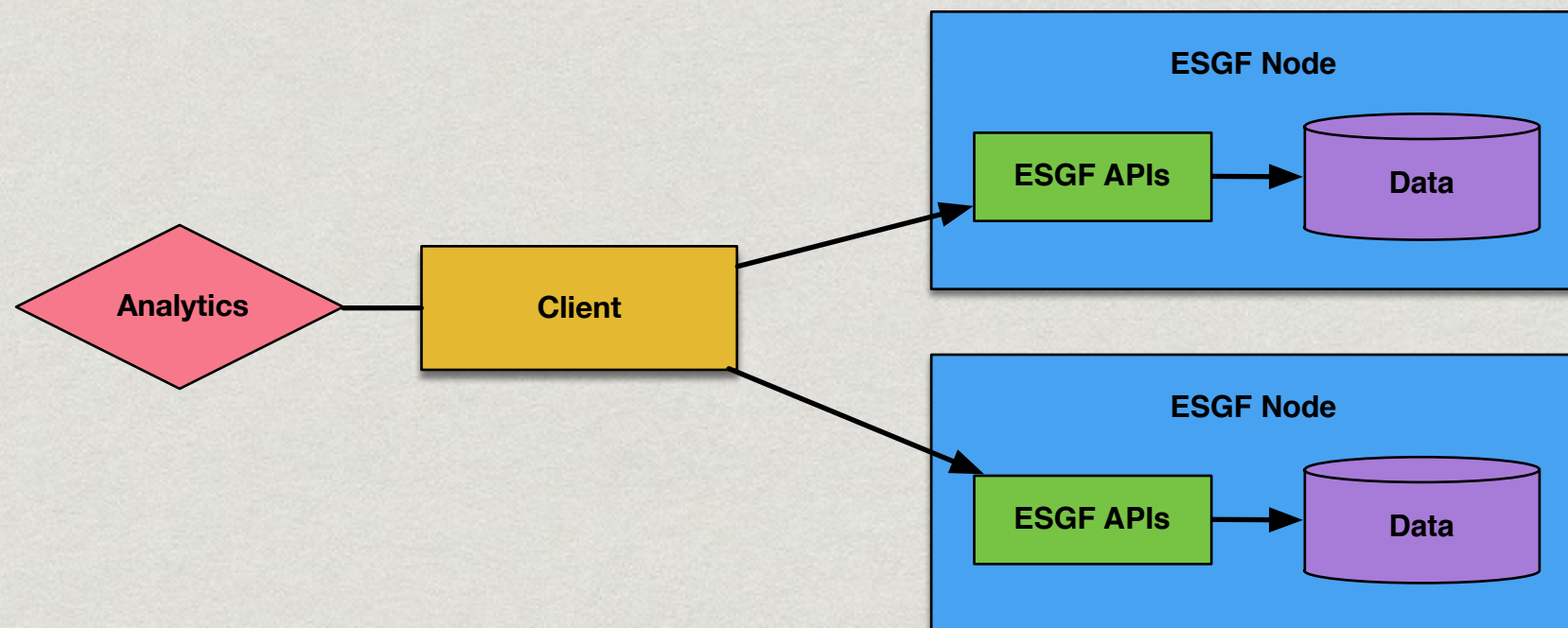


PART 3

**SUMMARY: USING ESGF AS A TESTBED
FOR ANALYTICAL DATA PROCESSING**

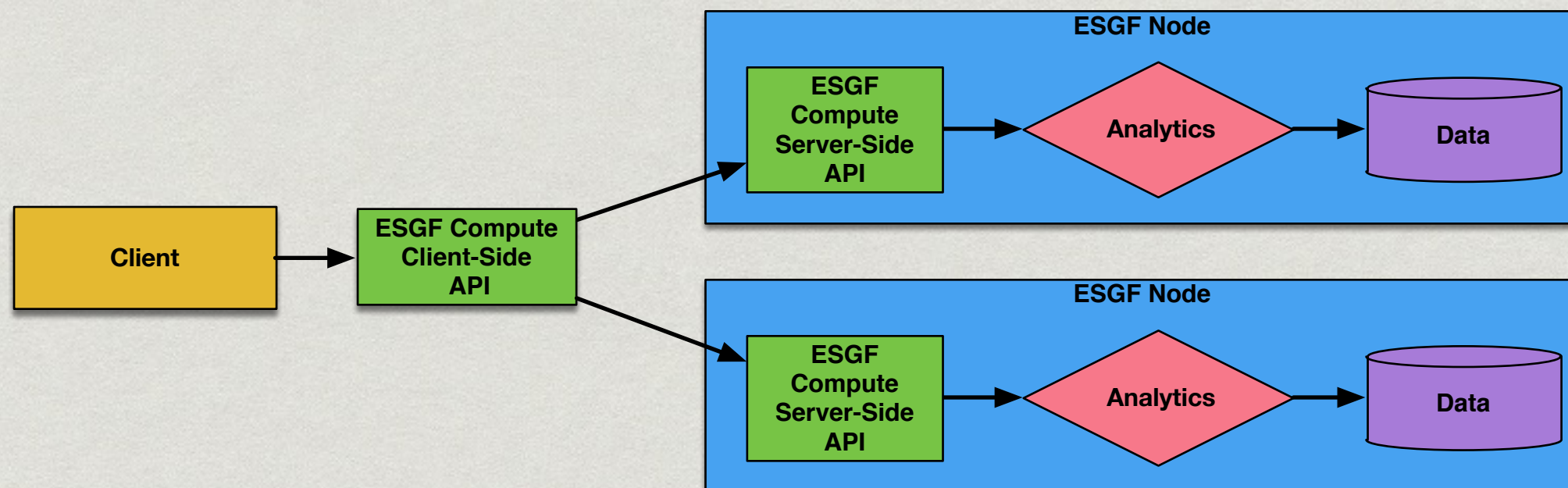
Client-Driven Architecture

- * A local client access all data via ESGF APIs (search, download, subset)
- * All processing executed locally (laptop, in-premise, cloud)
- * Still takes advantage of a distributed federated archive, standard data/metadata formats, programmatic access APIs
- * Possibly no access control for read-only operations
- * Can be executed right now



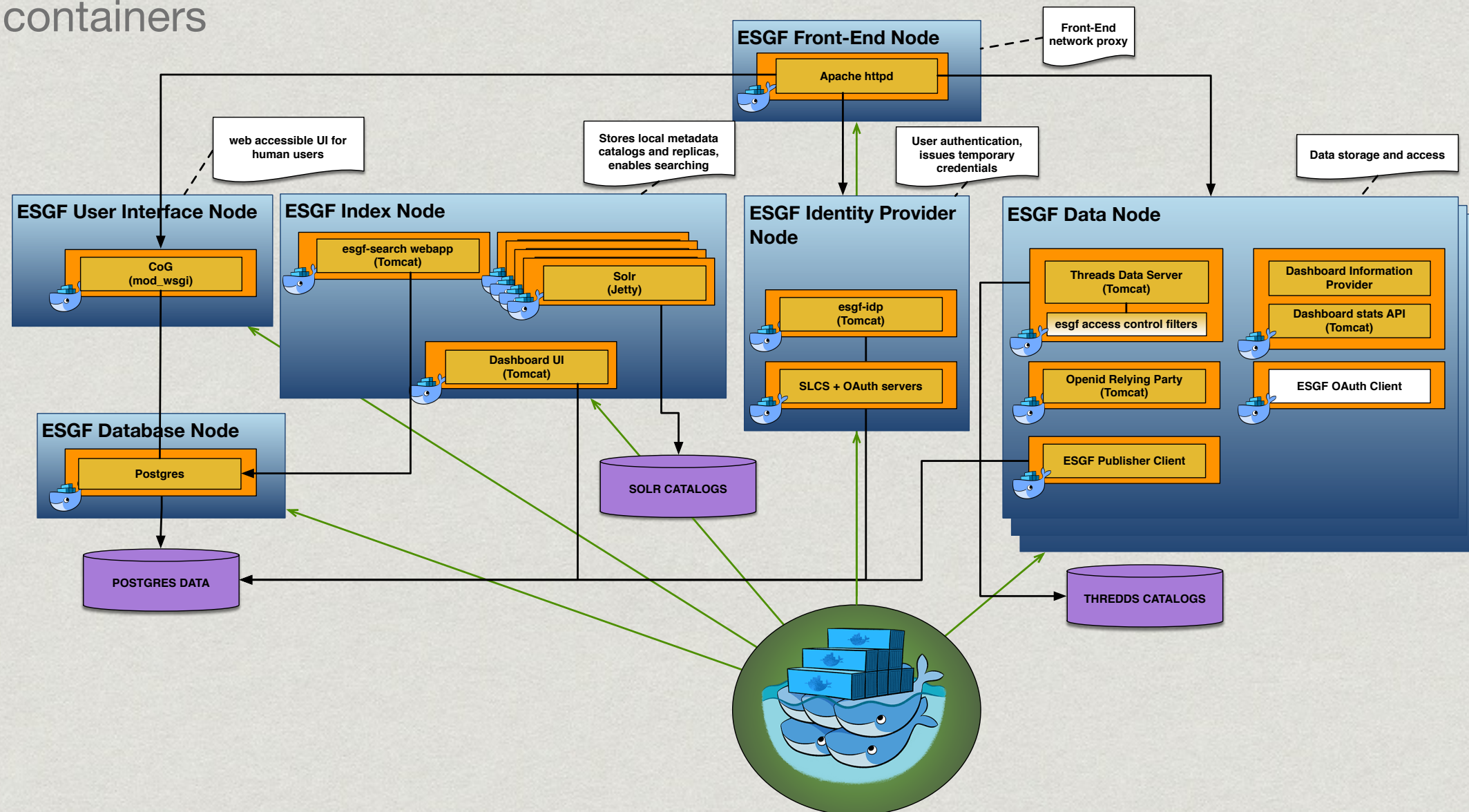
Server-Side Computation

- * In the near future (~12 months), ESGF will support server-side computation
- * Analytics can be executed close to the data, but must conform to the WPS API
- * Must use access control to authorize execution on remote servers
- * Open questions:
 - * How to deploy custom algorithms at each Node ?
 - * Maybe use a limited number of “friendly” Nodes
 - * How to execute workflows that span multiple Nodes ?



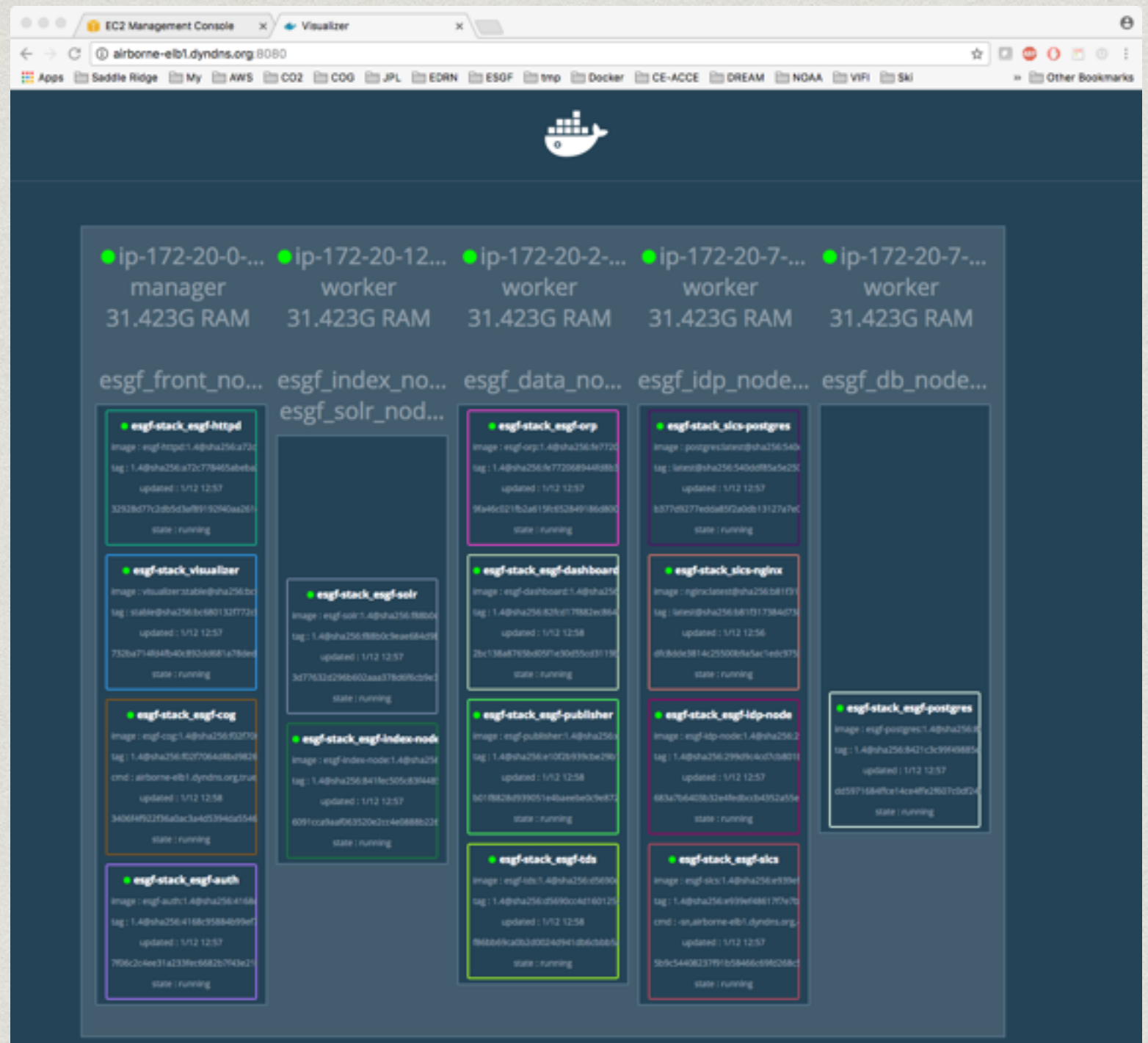
Containerization

- * ESGF is working on a new system architecture based on Docker containers
- * A “container” is a lightweight package that includes a program executable + all its dependencies + just enough OS to run it
- * In the future, all ESGF data services will be deployed and operated as Docker containers



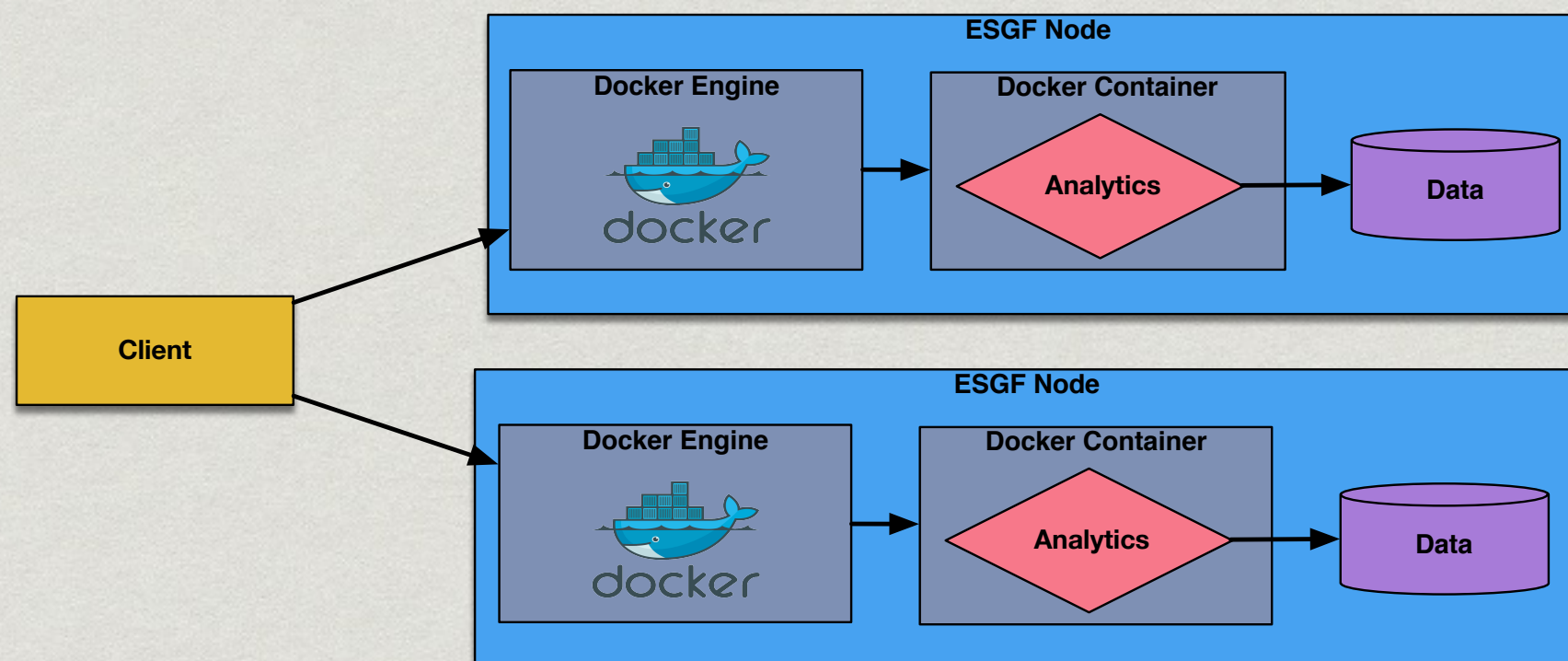
Containerization

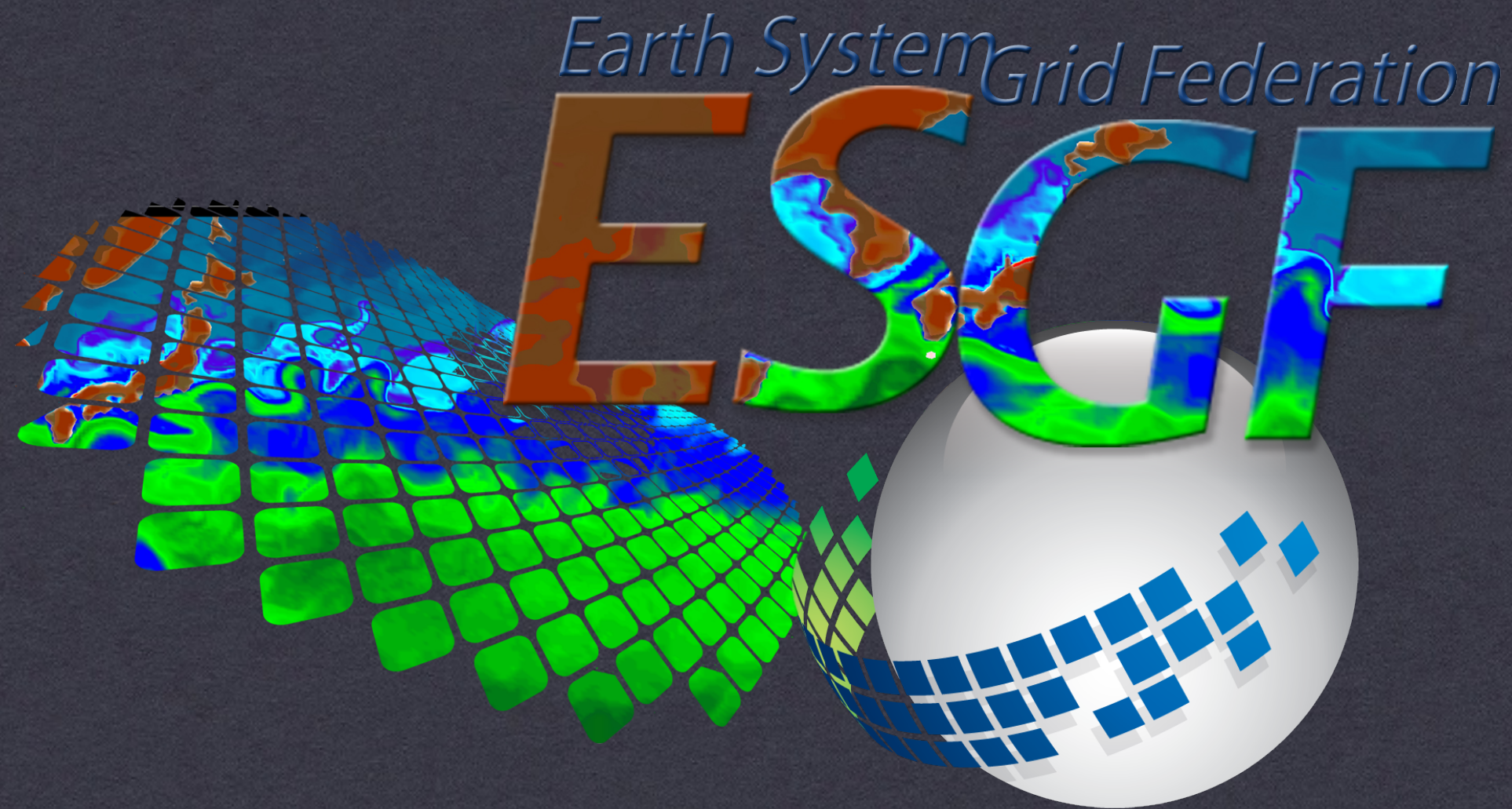
- * ESGF/Docker software stack 1.4 as deployed on an AWS cluster of 5 EC2 instances



Containerization

- * In the next few years, we can expect most ESGF sites to use Docker to operate at least some of their services - possibly the full ESGF stack
- * Docker engine at each site can be used to execute custom analytics as self-contained Docker containers
 - * No complex software installation at each site
 - * Direct read-only access to local data - no data level access control
- * Open questions:
 - * How do you orchestrate containers running at distributed sites ?
 - * How do you authorize containers to run at each site ?





QUESTIONS ?